

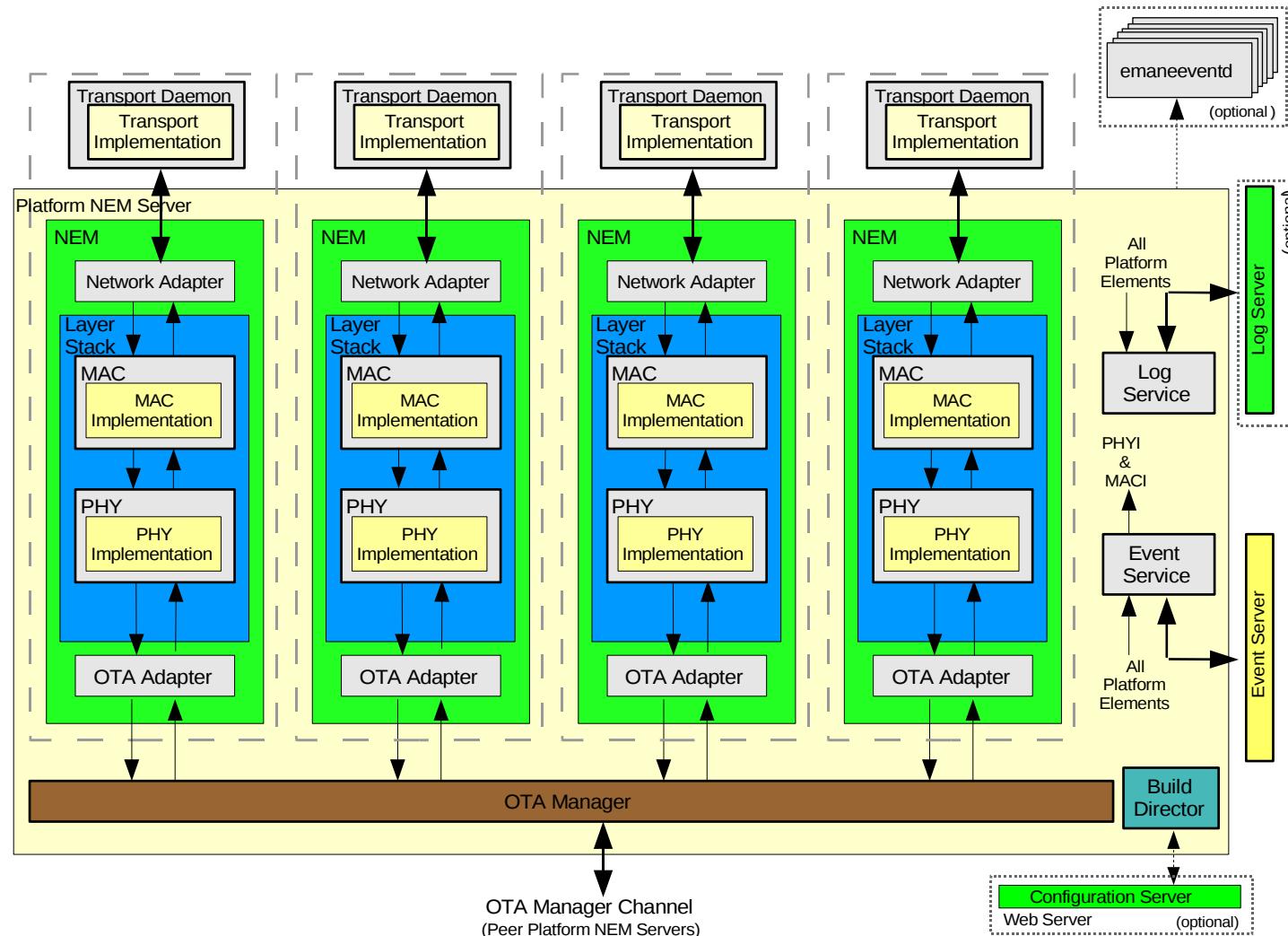
EMANE User Training

0.6.2

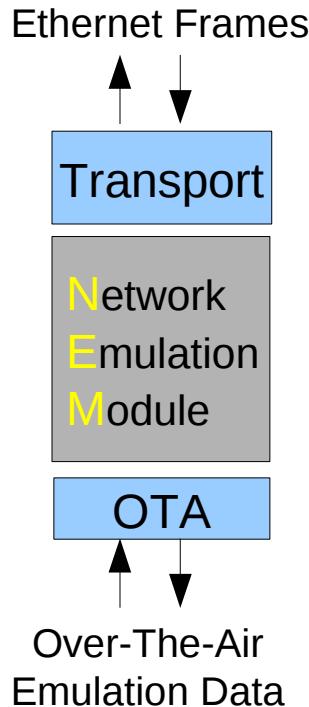
Extendable Mobile Ad-hoc Emulator

- Supports emulation of simple as well as complex network architectures
- Supports emulation of multichannel gateways
- Supports model specific configuration and control messaging
- Provides mechanisms to bridge emulation environment control information with non-emulation aware components
- Supports large scale testbeds with the same ease as small test networks
- Supports cross platform deployment (Unix, Linux, OSX, MS Windows)

EMANE Architecture



Emulation Stack Anatomy



- *Transport* – Mechanism responsible for transporting data to and from the emulation space. (Emulation/Application boundary interface)
- *Network Emulation Module (NEM)* - Emulation implementation functionality for a given radio model
- *Over-The-Air Manager* - Provides the mechanism NEMs use to communicate

Transport

- Realization of an emulation/application boundary interface.
Provides the entry and exit point for all data routed through the emulation.
 - For example the two ethernet frame transports: Virtual Transport and Raw Transport are responsible for interfacing with the underlying operating system
 - TunTap (linux, OS X)
 - WinTap (win32)
 - libpcap (linux, OS X)
 - winpcap (win32)
 - Supports:
 - IPv4
 - IPv6
 - Unicast
 - Broadcast
 - Multicast
 - Throughput limitation
- Tunnels transport data as opaque payload to the corresponding NEM.

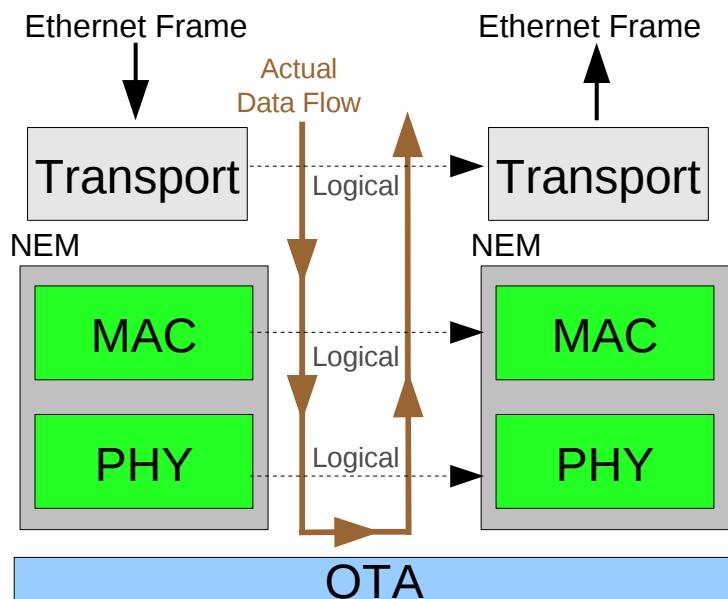
Network Emulation Module

- NEMs composed of two components
 - MAC Layer – Medium Access Control Layer emulation functionality
 - CSMA
 - TDMA & Hybrid schemes
 - Queue Management, Discard, QOS
 - Adaptive Transmission Protocols (Power and Data Rate)
 - PHY Layer – Physical Layer emulation functionality
 - Filter out of band packets
 - Waveform Timing
 - Half duplex operations
 - Transmit Power and antenna gain
 - Directional antenna support
 - Account for noise/collision
 - Probability of reception based on BER, RSSI, Packet Size

Over-The-Air Manager

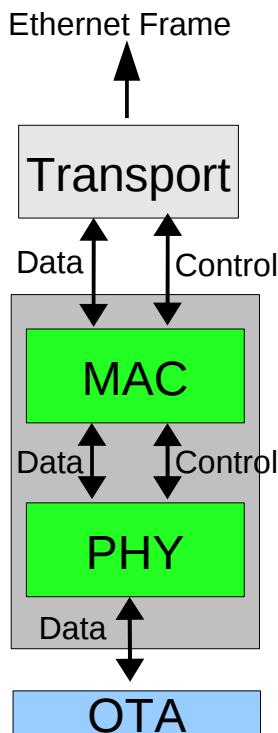
- Messaging infrastructure to deliver emulation radio model messages to all nodes in the deployment
 - Uses a multicast channel for message distribution for NEMs hosted by an NEM server different than that of the source NEM
 - Uses shared memory for message distribution for NEMs hosted by the same NEM server
- All messages are delivered to every NEM participating in the emulation (PHY Layer activity)
 - Provides the ability to model complex PHY phenomena such as RF interference
 - Multiple OTA multicast channels can be used to reduce overhead
 - OTA multicast channel can be disabled when utilizing a single NEM server

Emulation Stack Communication



- Ethernet Frames flow opaquely through the emulation stack
- Metadata is appended to support intra-layer communication as packets travel downstream between layers
- Metadata is stripped and processed as packets travel upstream between layers

Inter-Layer Communication



- Each stack layer has both a data and control path
- Control messages are only valid between contiguous layers
 - Examples of control message use:
 - Per packet RSSI (PHY to MAC)
 - Carrier Sense (PHY to MAC)
 - Transmission Control (MAC to PHY)
- Layer data destined for a corresponding layer uses the opaque data path for messaging
 - Data can be placed in a layer specific header of an existing downstream data message
 - A new data message may be generated just for layer specific messaging

PHY Layer Communication

- Radio PHY Models use a common header to allow cooperation between heterogeneous models in the same deployment
 - PHY Type
 - Source
 - Destination
 - Time
 - Message Duration
 - Center Frequency
 - Bandwidth
 - Data Rate
 - Antenna Gain
 - Antenna Beam Width
 - Antenna Elevation
 - Antenna Azimuth
 - Tx Power
- Data used to implement a higher fidelity RF model
 - Noise floor calculations
 - Frequency interference
 - Directional antenna support

Emulation Events

- Emulation data is distributed in realtime to NEMs by the [EMANE Event Service](#). Event data is distributed using the Event Multicast channel.
- Emulation components that generate events are called [Event Generators](#).
 - Events are distributed as opaque data
 - Only Event Generators and components subscribed to the specific events process the data
- Emulation event data is also available to non emulation components through the [EMANE Event Daemon](#)
 - An event agent plugin API exposes all transmitted events for external processing without exposing the mechanisms used to transmit the data

Existing Emulation Events

Component	Location	Pathloss	Comm Effect
RF Pipe *	●	●	
IEEE 802.11 abg		●	
Comm Effect			●
GPSd Location Agent	●		

* RF Pipe uses either Location Events or Pathloss Events
to determine pathloss based on configuration.



EMANE Application Suite

EMANE Applications

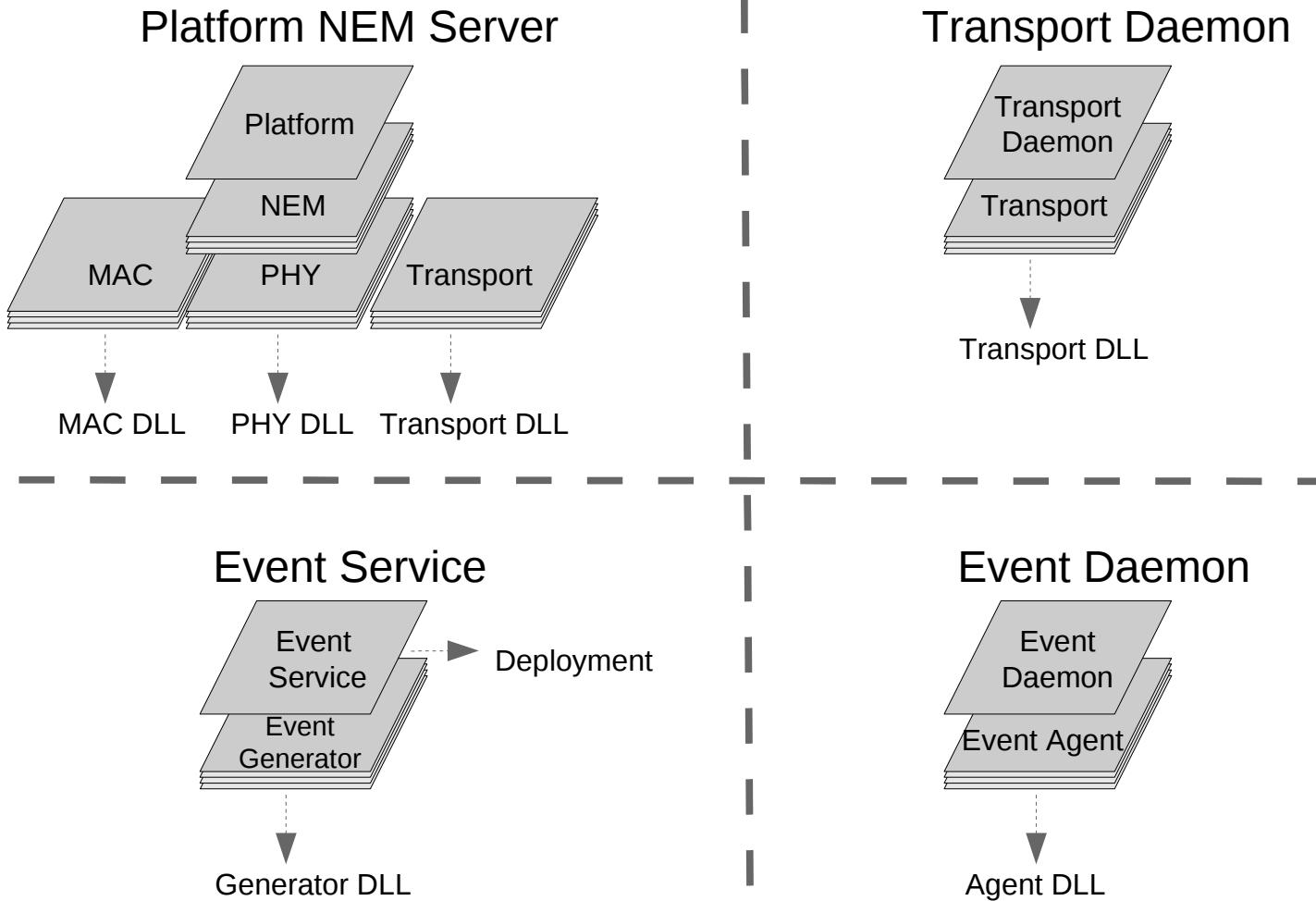
- *emane* – NEM Platform Server application. Creates and manages one or more NEMs.
Input XML: platform file
- *emanettransportd* – Transport Daemon application. Creates and manages one or more transports.
Input XML: transportdaemon file
- *emaneeventd* – Event Daemon application. Creates and manages one or more event agents.
Input XML: eventdaemon file
- *emaneeventservice* – Event Service application. Creates and manages one or more event generators.
Input XML: eventservice file

EMANE Utilities

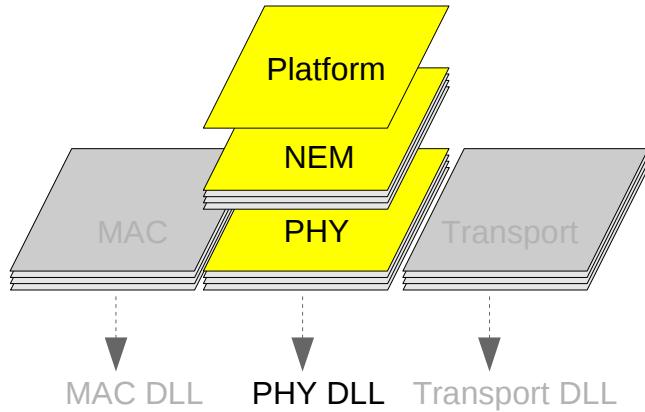
- *emanelevel* – Dynamical modify the log level of a running emane process
- *emanegentransportxml* – Generate corresponding transportdaemon XML from a platform XML file.
Input XML: platform file
- *emanegendeploymentxml* – Generate corresponding deployment XML from a platform XML file.
Input XML: platform file
- *emanegenxml* – Generate all XML required for an EMANE deployment from a properly formatted text input file
Input File: emanegenxml input file

EMANE XML Hierarchy

EMANE XML Hierarchy



EMANE XML Hierarchy Example



```
<phy name="RF-PIPE PHY" library="rfpipephylayer">
  <param name="txpower" value="0"/>
  <param name="antennagain" value="0"/>
  <param name="bitrate" value="0"/>
  <param name="jitter" value="0"/>
  <param name="frequency" value="2347"/>
  <param name="bandwidth" value="20"/>
  <param name="propagationdelay" value="0"/>
  <param name="pathlosssource" value="0"/>
  <param name="pathlossalgorithm" value="0"/>
  <param name="defaultconnectivity" value="on"/>
  <param name="rxpowernolossthreshold" value="90"/>
  <param name="rxpowerfulllossthreshold" value="110"/>
</phy>
```

```
<platform name="Platform 1" id="1">
  <param name="otamanagerchannelenable" value="off"/>
  <param name="eventservicegroup" value="224.1.2.8:45703"/>
  <param name="eventservicedevice" value="lo"/>
  <nem name="NODE-001" id="1" definition="rfpipenem.xml">
    <param name="platformendpoint" value="localhost:8181"/>
    <param name="transportendpoint" value="localhost:8171"/>
    <phy definition="rfpipephy.xml">
      <param name="frequency" value="3000"/>
    </phy>
    <transport definition="transraw.xml">
      <param name="device" value="eth1"/>
    </transport>
  </nem>
  <nem name="NODE-002" id="2" definition="rfpipenem.xml">
    <param name="platformendpoint" value="localhost:8182"/>
    <param name="transportendpoint" value="localhost:8172"/>
    <transport definition="transraw.xml">
      <param name="device" value="eth2"/>
    </transport>
  </nem>
```

```
<nem name="RF-PIPE NEM">
  <mac definition="rfpipemac.xml"/>
  <phy definition="rfpipephy.xml">
    <param name="frequency" value="3340"/>
  </phy>
  <transport definition="transraw.xml"/>
</nem>
```

Deployment Options

NEM Platform Server

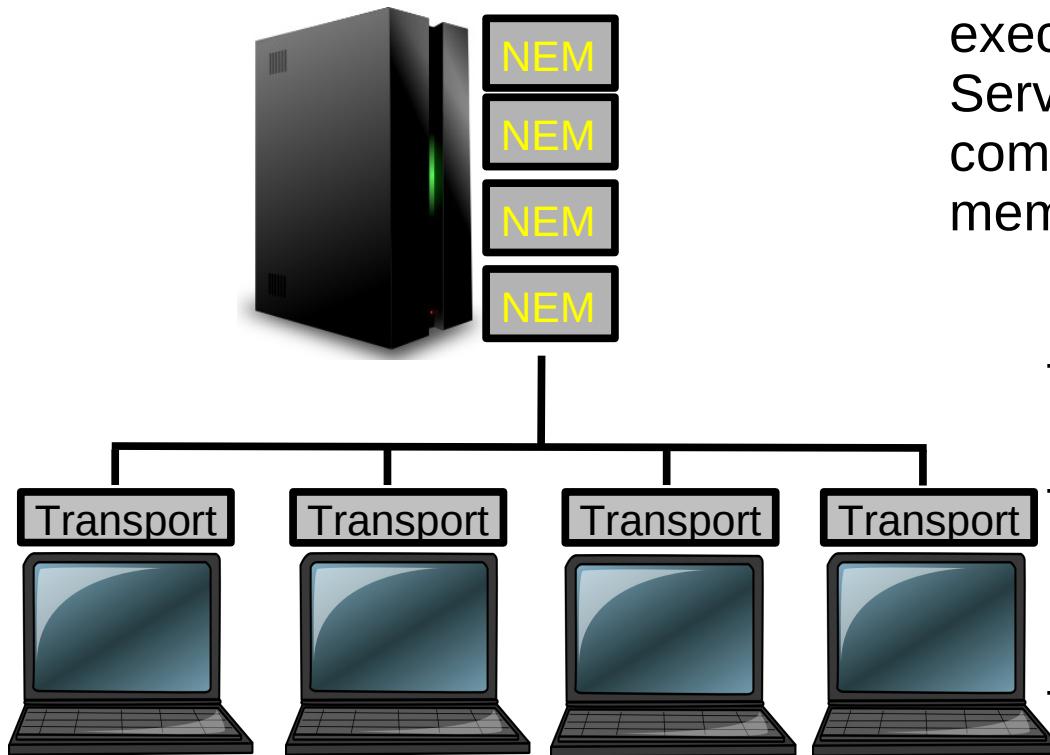
Emulation Deployment

There are three different configurations for EMANE deployments:

- **Centralized Mode** – All emulation is performed on a central server
- **Distributed Mode** – Emulation is distributed across test nodes
- **Hybrid Mode** – A mixture of Centralized Mode and Distributed Mode test nodes

Modes only define the relationship between NEMs and NEM Platform Servers in the deployment.

Centralized Deployment



All emulation functionality (NEM) is executed on a single NEM Platform Server. All OTA Manager communication is done using shared memory.

Pros:

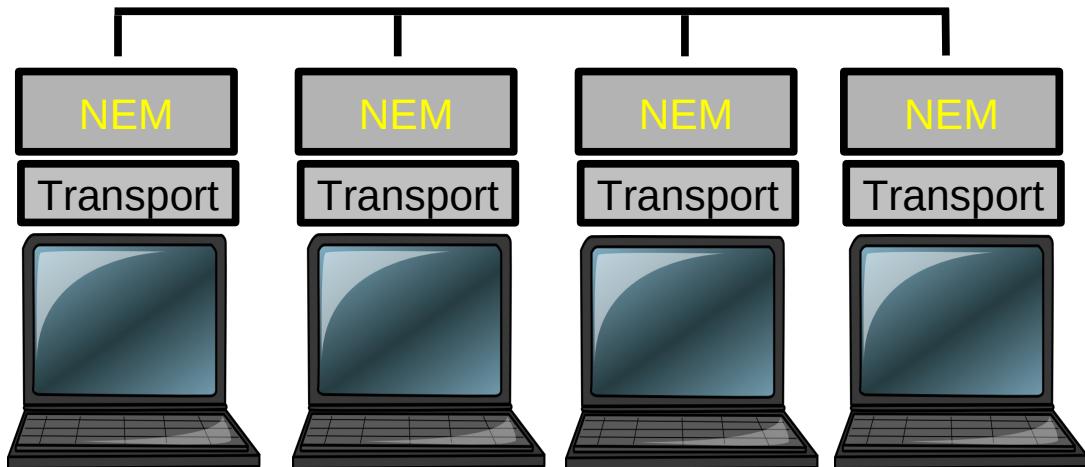
- Allows dedicated emulation processing
- Simplifies emulator control and configuration deployment

Cons:

- Larger centralized computing power required to emulate large networks

Distributed Deployment

Emulation functionality (NEM) distributed across test nodes. OTA Manager communication is done using a multicast channel.



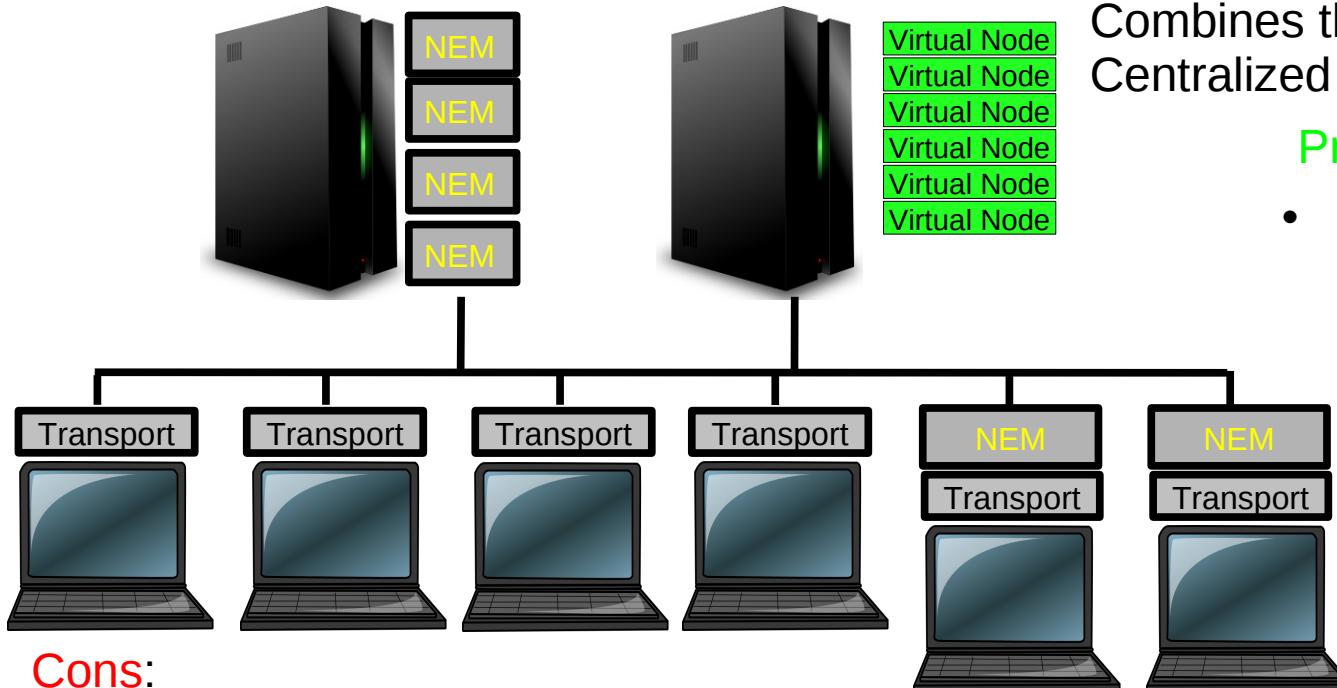
Pros:

- Distributes emulation functionality across all test nodes
- Easier to grow emulation deployment

Cons:

- Slightly more complex to control
- Timing accuracy must be addressed since emulation spans multiple RTCs

Hybrid Deployment



Combines the best features of Centralized and distributed mode

Pros:

- Flexibility of growing and shrinking emulation deployment as asset become available

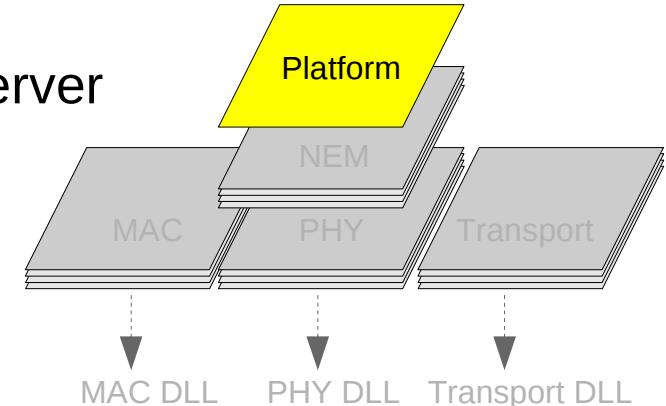
Cons:

- Complex to control
- Timing accuracy must be addressed

Platform XML Explained

Platform NEM Server XML is the root of all EMANE configuration

- Transport daemon XML and deployment XML automatically generated from platform XML
- Specify NEMs created by the platform server
 - NEM types (can be heterogeneous)
 - Associate transports
 - Transport Types
 - Endpoint connection information
- Specify configuration parameter values
 - OTA Manager multicast channel (group, port, and device)
 - Event multicast channel (group, port, and device)
 - Override values specified in the PHY, MAC, and transport XML



Platform XML Sample

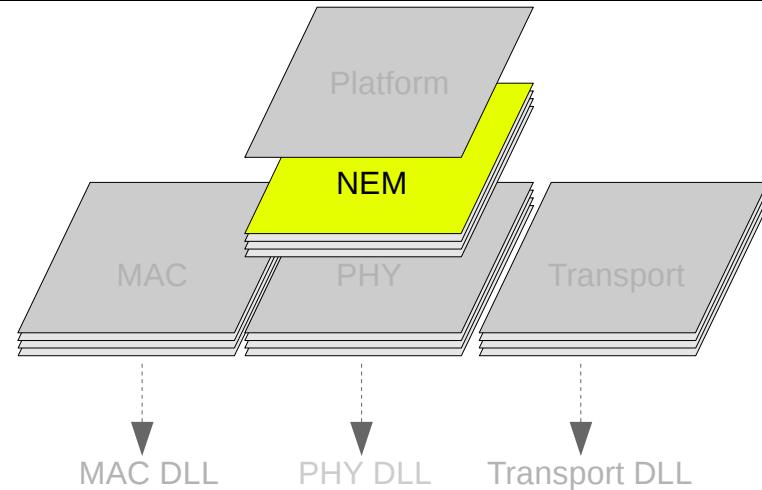
- Platform NEM Server (emane), transport daemon (emanettransportd), and event service (emaneeventservice) reside on the same machine
 - *platformendpoint* and *transportendpoint* using localhost
 - *eventservicedevice* using loopback
- NEMs using same definition:
rfpipenem.xml
- NEMs using *transraw.xml*
 - Overriding transport *device* configuration parameter to associate different devices
 - Using a transport *group* tag to combine into a single transport daemon

```

<platform name="Workstation 1" id="1">
    <param name="otamanagergroup" value="224.1.2.8:45702"/>
    <param name="eventservicegroup" value="224.1.2.8:45703"/>
    <param name="otamanagerdevice" value="eth0"/>
    <param name="eventservicedevice" value="lo"/>
    <nem name="NODE-001" id="1" definition="rfpipenem.xml">
        <param name="platformendpoint" value="localhost:8181"/>
        <param name="transportendpoint" value="localhost:8171"/>
        <transport definition="transraw.xml" group="single">
            <param name="device" value="eth1" />
        </transport>
    </nem>
    <nem name="NODE-002" id="2" definition="rfpipenem.xml">
        <param name="platformendpoint" value="localhost:8182"/>
        <param name="transportendpoint" value="localhost:8172"/>
        <transport definition="transraw.xml" group="single">
            <param name="device" value="eth2" />
        </transport>
    </nem>
</platform>
```

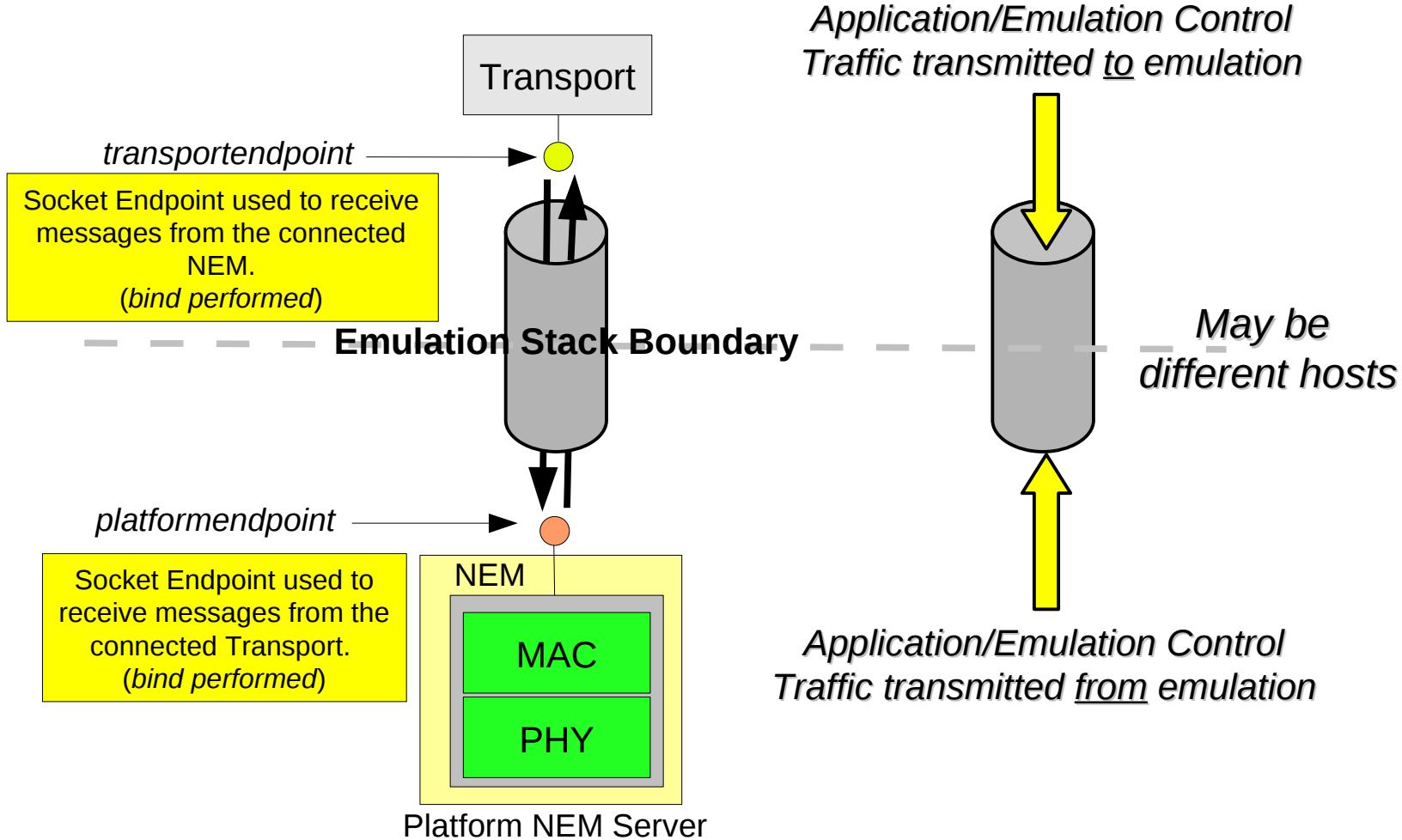
NEM XML Explained

- Specify NEM components
NEM Components must match any specified in platform XML
 - PHY Layer XML
 - MAC Layer XML
 - Transport XML
- Specify configuration parameter values
 - Override values specified in the PHY, MAC, and transport XML
 - Values can be overridden by those specified in the platform XML



```
<nem name="RF-PIPE NEM">
  <mac definition="rfpipemac.xml"/>
  <phy definition="rfpipephy.xml"/>
  <transport definition="transraw.xml"/>
</nem>
```

platformendpoint and *transportendpoint* Explained



Workshop 1 – Centralized EMANE Deployment

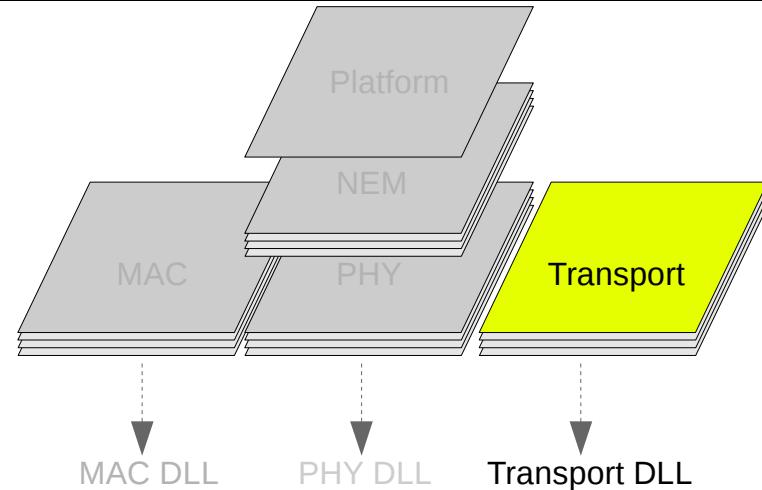
Workshop 2 – Distributed EMANE Deployment

Workshop 3 – Hybrid EMANE Deployment

Transports

Transport XML Explained

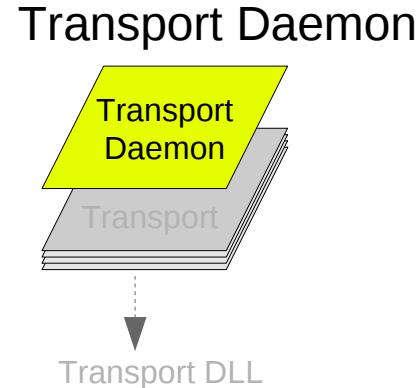
- library attribute specifies Transport DLL
 - libtransvirtual.so (Linux and OS X)
 - libtransvirtual.dll (win32)
- Specify configuration parameter defaults
 - Values can be overridden by those specified in the NEM, platform, and transport daemon XML files



```
<transport name="Virtual Transport" library="transvirtual">
  <param name="bitrate" value="0"/>
  <param name="devicepath" value="/dev/net/tun"/>
  <param name="device" value="emane0"/>
  <param name="address" value="172.30.1.1"/>
  <param name="mask" value="255.255.0.0"/>
</transport>
```

Transport Daemon XML Sample

- Automatically generated from the platform XML using emanegenttransportxml utility
- Specify the transport instances created by the transport daemon
 - Transport types
 - Associate NEMs
 - Ids
 - Endpoint connection information
 - Specify transport configuration parameter values



```
<transportdaemon>
    <instance nemid="1">
        <param name="transportendpoint" value="localhost:8171"/>
        <param name="platformendpoint" value="localhost:8181"/>
        <transport definition="transraw.xml">
            <param name="device" value="eth1"/>
        </transport>
    </instance>
    <instance nemid="2">
        <param name="transportendpoint" value="localhost:8172"/>
        <param name="platformendpoint" value="localhost:8182"/>
        <transport definition="transraw.xml">
            <param name="device" value="eth2"/>
        </transport>
    </instance>
</transportdaemon>
```

Virtual Transport

- Creates a virtual interface on a test node. All traffic routed to that interface is encapsulated and tunneled to/from the test node's respective NEM.
 - Ideal when you are capable of running emulation software on the network device
 - Only requires a single NIC per network device
 - Virtual device configuration support
 - IPv4 and IPv6 capable

See Manual: *transvirtual.pdf*

Raw Transport

- Opens an existing interface in promiscuous mode. All traffic received on that interface is encapsulated and tunneled to/from the test node's respective NEM.
 - Ideal when the network device using the emulator is a black box (not capable of running emulation software)
 - Requires more dedicated hardware – physical interface per NEM required
 - IPv4 and IPv6 capable

See Manual: *transraw.pdf*

Interactive Transport

- Each instance implements a telnet server that provides a testing shell which supports transport to transport unicast and broadcast *pinging* with simple statistics
 - Supports simple test cases and training activities
 - Supported commands:
 - clear
 - quit
 - whoami
 - exit
 - stats
 - ping
 - version

See Manual: *emaentransinteractive.pdf*

Workshop 4 – Virtual Transport Configuration

Workshop 5 – Raw Transport Configuration

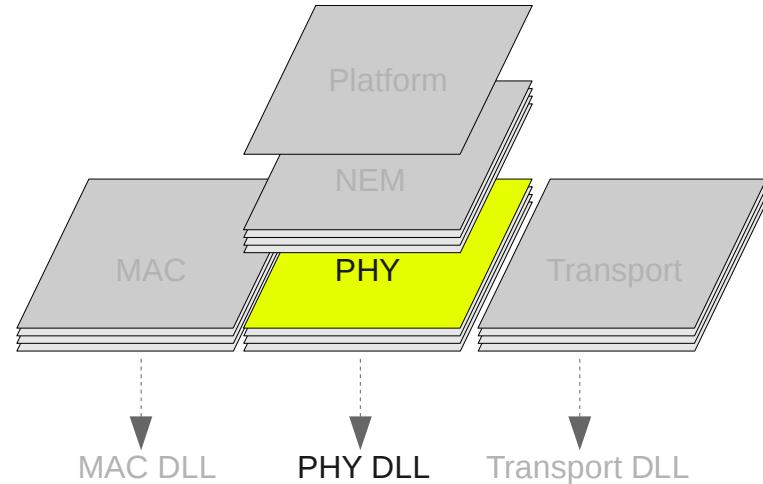
Workshop 6 – Interactive Transport Configuration

Radio Models

Network Emulation Models

PHY XML Explained

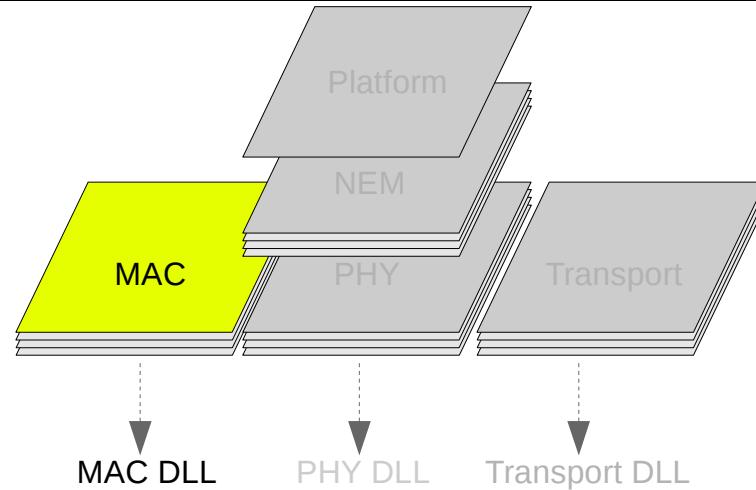
- library attribute specifies PHY DLL
 - librpipephylayer.so (Linux and OS X)
 - librpipephylayer.dll (win32)
- Specify configuration parameter defaults
 - Values can be overridden by those specified in the NEM and platform XML files



```
<phy name="RF-PIPE PHY" library="rpipephylayer">
    <param name="txpower" value="0"/>
    <param name="antennagain" value="0"/>
    <param name="bitrate" value="0"/>
    <param name="jitter" value="0"/>
    <param name="frequency" value="2347"/>
    <param name="bandwidth" value="20"/>
    <param name="propagationdelay" value="0"/>
    <param name="pathlosssource" value="0"/>
    <param name="pathlossalgorithm" value="0"/>
    <param name="defaultconnectivity" value="on"/>
    <param name="rxpowernolossthreshold" value="90"/>
    <param name="rxpowerfulllossthreshold" value="110"/>
</phy>
```

MAC XML Explained

- library attribute specifies MAC DLL
 - libieee80211abgmaclayer.so (Linux and OS X)
 - libeee80211abgmaclayer.dll (win32)
- Specify configuration parameter defaults
 - Values can be overridden by those specified in the NEM and platform XML files



```
<mac name="IEEE802.11" library="ieee80211abgmaclayer">
  <param name="enablepromiscuousmode" value="off"/>
  <param name="distance" value="1000"/>
  <param name="rtsthreshold" value="0"/>
  <param name="mode" value="0"/>
  <param name="unicastrate" value="0"/>
  <param name="multicastrate" value="0"/>
  <param name="wmmenable" value="off"/>
  <param name="queuesize" value="0:255"/>
  <param name="retrylimit" value="0:3"/>
  <param name="cwmin" value="0:31"/>
</mac>
```

RF Pipe Model

- Generic radio model with highly configurable PHY Layer that can provide a low fidelity emulation for a variety of waveforms
 - Features include:
 - Out of band packet filtering
 - Externally computed pathloss model with realtime dissemination
 - Realtime pathloss computations using propagation models
 - Free space
 - 2-ray flat earth
 - PoR utilizing Receive Power and uniform linear packet loss distribution based on min and max receive power thresholds
 - Transmission delay using bit rate, propagation delay, and jitter configuration parameters

See Manual: *rfpipe.pdf*

IEEE 802.11 abg Model

- Emulates IEEE 802.11 MAC layer's Distributed Coordination Function (DCF) channel access scheme on top of the IEEE 802.11 Direct Spread Spectrum Sequence (DSS) and Orthogonal Frequency Division Multiplexing (OFDM)
 - MAC Layer features include:
 - 802.11b (DSS rates: 1, 2, 5.5 and 11 Mbps)
 - 802.11a/g (OFDM rates: 6, 9, 12, 18, 24, 36, 48 and 54 Mbps)
 - 802.11b/g (DSS and OFDM rates)
 - DCF channel access
 - Unicast and broadcast (RTS/CTS/ACK capable)

IEEE 802.11 abg Model

- PHY Layer features include:
 - Out of band packet filtering
 - Externally computed pathloss model with realtime dissemination
 - Half duplex operation
 - PoR utilizing Bit Error Rate (BER) curves based on SINR, packet size and data rate

See Manual: *ieee80211abg.pdf*

Comm Effect Model

- Network Emulation Model that allows detailed control of individual links
 - Features:
 - Externally computed effect model with realtime dissemination
 - Asymmetric link probability of loss
 - Asymmetric link probability of duplication
 - Asymmetric latency specification
 - Asymmetric jitter specification
 - Packet filtering (Source, Destination, TOS/DSCP, Protocol)

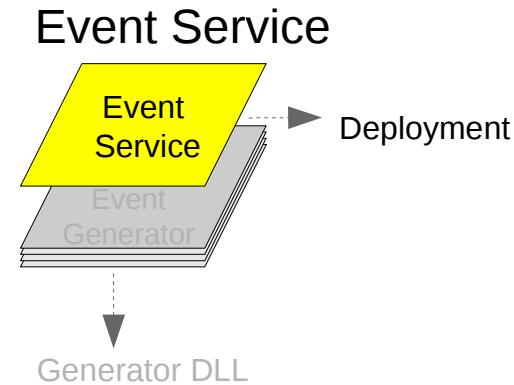
See Manual: *commeffect.pdf*

Workshop 7 – RF Pipe Model
Workshop 8 – IEEE 802.11 abg Model
Workshop 9 – Comm Effect Model

Event Generators

Event Service XML Explained

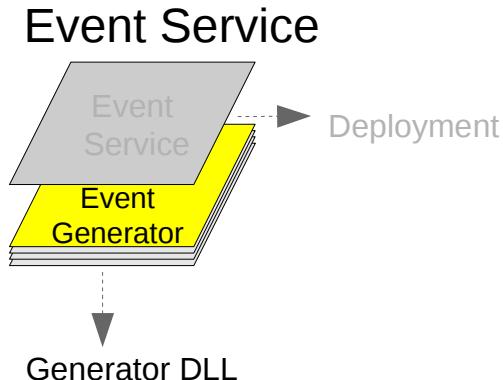
- Specify the Event Generators created by the event service
 - List generator XML
 - *mitremobilitygenerator.xml*
- Specify deployment XML file containing emulation NEM platform relationships
 - *deployment.xml*
- Specify configuration parameter values
 - Event multicast channel (group, port, and device)



```
<eventservice name="Sample Event Service" deployment="deployment.xml">
  <param name="eventservicegroup" value="224.1.2.8:45703"/>
  <param name="eventservicedevice" value="eth0"/>
  <generator name="Mobility Model Generator"
             definition="mitremobilitygenerator.xml" />
</eventservice>
```

Event Generator XML Explained

- library attribute specifies Generator DLL
 - libmitremobilitymodelgenerator.so (Linux and OS X)
 - libmitremobilitymodelgenerator.dll (win32)
- Specify configuration parameter defaults
 - Values can be overridden by those specified in the event service XML file



```

<eventgenerator name="mitremobilitymodel"
  library="mitremobilitygenerator">
  <param name="inputfileformat" value="matrix%02d.txt"/>
  <param name="inputfilecount" value="11"/>
  <param name="totalnodes" value="70"/>
  <param name="maxnemidpresent" value="12"/>
  <param name="repeatcount" value="1"/>
  <param name="utmzone" value="18T"/>
  <param name="entryreplay" value="0:3 1:10 2:100"/>
</eventgenerator>

```

Mitre Mobility Model Generator

- Creates pathloss and location events from input files in the Mitre Mobility Format
- The mobility input files contain pathloss and location information between MANET nodes on one second boundaries.
 - Supports symmetric and asymmetric pathloss
- Supports playback of specific time indexes for configurable durations

See Manual: *mitremobility.pdf*

Emulation Script Generator

- Creates location events from input files in the NRL Emulation Script Format
- Emulation Script Format is an XML file containing event elements.
 - Each element contains two child elements:
 - Time - specifying the amount of time that has elapsed since the previous event.
 - Node - specifying the id (using an attribute) and the location (using a child element) of a given node in the network.

See Manual: *emulationscript.pdf*

Emulation Event Log Generator

- The Emulation Event Log (EEL) generator creates EMANE events from input files in EEL Format. EEL format was developed by the Protean Research Group at Naval Research Laboratory.
 - Supports generating Pathloss and Location events

See Manual: *emanegeneel.pdf*

See: *EmulationScript Schema Description Protean Research Group,
Naval Research Laboratory Code 5522*

Workshop 10 – Mitre Mobility Event Generator

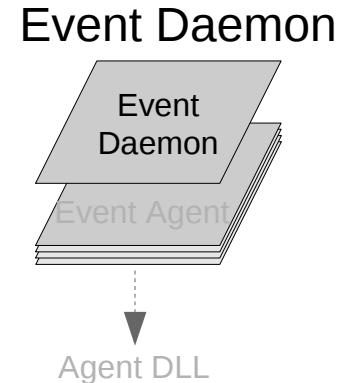
Workshop 11 – Emulation Script Event Generator

Workshop 12 – Emulation Event Log Generator

Event Daemon Agents

Event Daemon XML Explained

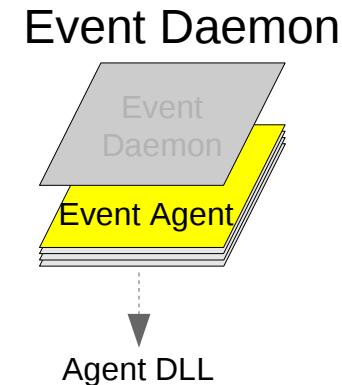
- Specify the Event Agents created by the event daemon
 - List Agent XML
 - gpdlocationagent.xml
- Specify the NEM id to associate with the event agents
 - Not used by all event agents (set to 0)
- Specify configuration parameter values
 - Event multicast channel (group, port, and device)



```
<eventdaemon name="EMANE Event Daemon" nemid="1">
    <param name="eventservicegroup" value="224.1.2.8:45703"/>
    <param name="eventservicedevice" value="eth0"/>
    <agent definition="gpdlocationagent.xml"/>
</eventdaemon>
```

Event Agent XML Explained

- library attribute specifies event agent DLL
 - libgpslocationagent.so (Linux and OS X)
 - No win32 Support
- Specify configuration parameter defaults
 - Values can be overridden by those specified in the event daemon XML



```
<eventagent name="gpsdlocationagent"
            library="gpsdlocationagent">
    <param name="gpsdconnectionenabled" value="off"/>
    <param name="pseudoterminalfile"
          value="/tmp/gpsdlocation.pty"/>
    <param name="gpsdcontrolsocket"
          value="/tmp/gpsd.control"/>
</eventagent>
```

GPSd Location Agent

- Subscribes to location events and translates position information into NMEA strings which are written to a pseudo terminal device
 - Any NMEA aware application capable of reading from a serial port can open the pseudo terminal and process the NMEA strings
 - Can be configured to automatically connect to a running gpsd process

See Manual: *gpsdlocation.pdf*

Workshop 13 – GPSd Location Agent

Tips

EMANE File Reference Chart

Platform XML	NEM XML*	MAC XML	PHY XML	Shim XML	Transport XML
Required	Required	Required	Required	Required	Required
NEM XML Transport XML	MAC XML PHY XML	MAC DLL	PHY DLL	Shim DLL	Transport DLL
Optional PHY XML MAC XML Shim XML	Optional Shim XML				

Application: emane

Transport Daemon XML	Transport XML
Required Transport XML	Required Transport DLL

Application: emanetransportd

Event Service XML	Event Generator XML
Required Event Generator XML Deployment XML	Required Event Generator DLL

Application: emaneeventservic

Event Daemon XML	Event Agent XML
Required Event Agent XML	Required Event Agent DLL

Application: emaneeventd

* Structured NEM only

Tips

- All EMANE XML can be validated prior to executing the respective application:

```
> xmllint --valid --noout platform.xml
```
- The following applications will daemonize when launched with --loglevel 0
 - emane
 - emanettransportd
 - emaneeventd
 - emaneeventservice
- All emane applications respond to SIGQUIT and SIGINT for shutdown

Tips

- EMANE applications can use local or remote XML
 - Remote XML configuration can simplify large distributed deployments
> emane <http://configserver/test1/platform1.xml> --logl 4
- Use the emanecreatdtdpath utility to modify the dtd locations of your XML and DTDs
> emanecreatdtdpath –dtdpath http://configserver/dtd *.xml *.dtd
- Launch applications with --loglevel 4 to see verbose developer logs for debugging purposes
- Configure devices dedicated to Raw Transports with atypical addresses and full netmasks to highlight they should not be used for direct communication

Tip - Multichannel Gateway

- A multichannel gateway is created using multiple NEMs and transport instances and by configuring the appropriate routing
 - 1 NEM per channel
 - 1 Transport per channel
- Recommendations
 - NEMs should be part of the same platform NEM Server
 - Transport instances should be part of the same transport daemon
- Each gateway NEM (channel) should have PHY Layer separation
 - example: different frequency assignments or different NEM types
- Each NEM must have a unique id
 - Use the NEM name attribute to convey gateway ownership

```
<nem name="NODE-001 CHAN 2" id="10" definition="wipnnem.xml">
```

Tip - Multichannel Gateway

- Two Channel gateway
 - Chan 1 – 250MHz
 - Chan 2 – 350 MHz
- Name attribute used to provide gateway ownership
 - NODE-001 CHAN 1
 - NODE-001 CHAN 2
- NEM Ids are unique:
 - 1
 - 10

```
<platform name="Platform 1" id="1">
  <param name="otamanagergroup" value="224.1.2.8:45702"/>
  <param name="eventservicegroup" value="224.1.2.8:45703"/>

  <nem name="NODE-001 CHAN 1" id="1" definition="wipnnem.xml">
    <param name="platformendpoint" value="localhost:8181"/>
    <param name="transportendpoint" value="localhost:8171"/>
    <phy definition="rfpipephy.xml">
      <param name="frequency" value="250"/>
    </phy>
    <transport definition="transraw.xml" group="single1">
      <param name="device" value="eth5" />
    </transport>
  </nem>

  <nem name="NODE-001 CHAN 2" id="10" definition="wipnnem.xml">
    <param name="platformendpoint" value="localhost:8190"/>
    <param name="transportendpoint" value="localhost:8160"/>
    <phy definition="rfpipephy.xml">
      <param name="frequency" value="350"/>
    </phy>
    <transport definition="transraw.xml" group="single1">
      <param name="device" value="eth10" />
    </transport>
  </nem>
</platform>
```

Tip -XML Auto Generation

- Most EMANE XML can be autogenerated using one of two methods:
 - Define the platform XML files and use **emanegentransportxml** and **emanegendeploymentxml**
 - See man page: [*emanegentransportxml*](#)
 - See man page: [*emanegendeploymentxml*](#)
 - Use **emanegenxml** and create a text input file describing the deployment
 - See man page: [*emanegenxml*](#)

Event Model Generation

EMANE Loss Controller

(For use with Mitre Mobility Model Generator)

Create New Mobility Model

Enter the parameters for the new model

Model Name:	<input type="text" value="My New Model"/>
Node Entry Count:	<input type="text" value="10"/>
Min Pathloss:	<input type="text" value="5"/>
Max Pathloss:	<input type="text" value="85"/>
Event Service IP:	<input type="text" value="224.1.2.8"/>
Event Service Port:	<input type="text" value="45703"/>
Event Service NIC:	<input type="text" value="eth1"/> <input type="button" value="▼"/>

EMANE Loss Controller [My New Model]

File Edit Help

Entry ID:	<input type="text" value="0"/>
Min Pathloss:	<input type="text" value="5"/>
Max Pathloss:	<input type="text" value="85"/>
Duration (sec):	<input type="text" value="1"/>
<input type="button" value="Fill All Min"/> <input type="button" value="Fill All Max"/>	
<input type="button" value="Quick Fill"/> <input type="button" value="All Nodes"/> <input type="button" value="▼"/>	

Total Entries: 1 Duration: 1

EMANE Comm Effect Controller

(For use with the Comm Effect Shim)

Create New Comm Effect Project

Enter the parameters for the new project

Project Name:	My Project
Node Entry Count:	10
Event Channel IP:	224.1.2.8
Event Channel Port:	45703
Event Channel NIC:	lo

OK Cancel

Enter Comm Effect Values

Node 4 to Node 6	Node 6 to Node 4
Latency: 5 <input type="text"/> secs 0 <input type="text"/> usecs 	Latency: 10 <input type="text"/> secs 0 <input type="text"/> usecs 
Jitter: 1 <input type="text"/> secs 0 <input type="text"/> usecs 	Jitter: 1 <input type="text"/> secs 0 <input type="text"/> usecs 
Loss: 0 <input type="text"/> % 	Loss: 0 <input type="text"/> % 
Duplicate: 0 <input type="text"/> % 	Duplicate: 0 <input type="text"/> % 

OK Cancel

Comm Effect Controller [My Project] *

File Edit Help

		2	3	4	5	6	7	8	9	10
Entry ID:	0	0	0	0	0	0	0	0	0	
Duration (sec):	1	0	0	0	0	0	0	0	0	
Quick Fill Panel										
Latency:	<input type="text"/> secs	<input type="text"/> usecs	<input type="checkbox"/>							
Jitter:	<input type="text"/> secs	<input type="text"/> usecs	<input type="checkbox"/>							
Loss:	<input type="text"/> %	<input type="checkbox"/>								
Duplicate:	<input type="text"/> %	<input type="checkbox"/>								
Apply all <input type="checkbox"/>										
Quick Fill		All Nodes								
Display Mode: latency <input type="button" value="▼"/>										
New Duplicate Delete << PREV NEXT >> Enter Playback Mode										
Total Entries: 1 Total Duration: 1										

Playback Mode

Comm Effect Controller [My Project] *

File Edit Help

Entry ID: 2		Display Mode: latency ▾								
Duration (sec): 10		2	3	4	5	6	7	8	9	10
1	15	15	15	15	15	15	15	15	15	15
2		15	15	15	15	15	15	15	15	15
3			15	15	15	15	15	15	15	15
4				15	15	15	15	15	15	15
5					15	15	15	15	15	15
6						15	15	15	15	15
7							15	15	15	15
8								15	15	15
9									15	15

Quick Fill Panel

Latency: 15 secs 0 usecs

Jitter: secs usecs

Loss: %

Duplicate: %

Apply all

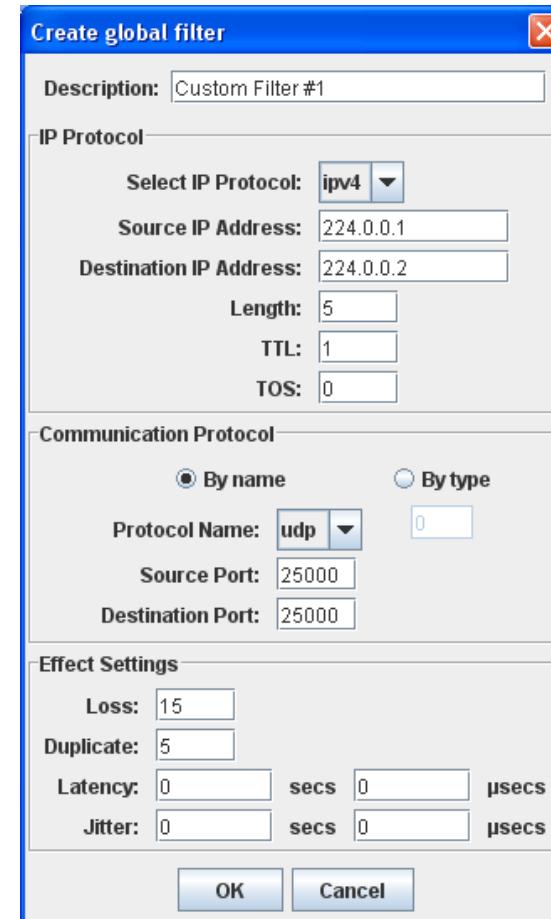
Quick Fill **All Nodes** ▾

Display Mode: Entry ID Time **Pause** **Stop** **Loop** **Enter Edit Mode**

0:00:00 0:00:35

Current Iteration: 1 Current Entry: 2 Current Time: 0:00:16

Filters



Workshop 14 – EMANE Loss Controller

Workshop 15 – EMANE Comm Effect Controller

Additional Deployment Options

Workshop 16 – Surrogate Radio Definitions

Event Service Language Bindings

Event Service Language Bindings

- **libemaneeventservice** – C Language API
 - Perl: EMANE::EventService
 - Python: emaneeventservice.EventService
- **libemaneeventpathloss** – C Language API
 - Perl: EMANE::Event::Pathloss
 - Python: emaneeventpathloss.EventPathloss
- **libemaneeventlocation** – C Language API
 - Perl: EMANE::Event::Location
 - Python: emaneeventlocation.EventLocation
- **libemaneeventcommeffect** – C Language API
 - Perl: EMANE::Event::CommEffect
 - Python: emaneeventcommeffect.EventCommEffect

Workshop 17 – libemaneeventservice
Workshop 18 – Perl bindings
Workshop 19 – Python bindings

Collaborative MANET Deployment

Putting it all together

- Workshop 20 – Distributed RF Pipe MANET**
- Workshop 21 – Centralized RF Pipe MANET**
- Workshop 22 – Distributed CommEffect MANET**
- Workshop 23 – Centralized CommEffect MANET**
- Workshop 24 – Centralized IEEE802.11abg MANET**

Where to go from here

More Information

- EMANE Homepage
<http://labs.cengen.com/emane>
- EMANE Users Mailing List
<http://pf.itd.nrl.navy.mil/mailman/listinfo/emane-users>
- EMANE Training
 - EMANE User Training (Duration: 1 day)
 - EMANE Developers Training (Duration: 3 days)
- EMANE Professional Services Available
Contact: labs@cengen.com